UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/560,203	12/08/2005	Stephen James Todd	GB920030063US1	3212

45112 7590 03/11/2008
Kunzler & McKenzie
8 EAST BROADWAY
SUITE 600
SALT LAKE CITY, UT 84111

EXAMINER	
BROPHY, MATTHEW J	

ART UNIT	PAPER NUMBER
2191	

MAIL DATE	DELIVERY MODE
03/11/2008	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Office Action Summary	**Application No.** 10/560,203	**Applicant(s)** TODD, STEPHEN JAMES	
	Examiner MATTHEW J. BROPHY	**Art Unit** 2191	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE _3_ MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on _08 December 2005_.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) _1-20_ is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) _1-20_ is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☒ The specification is objected to by the Examiner.

10)☒ The drawing(s) filed on _08 December 2005_ is/are: a)☐ accepted or b)☒ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☒ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☒ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☒ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)	4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____ .
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)	5) ☐ Notice of Informal Patent Application
3) ☒ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date _12/8/2005_.	6) ☐ Other: _____.

DETAILED ACTION

Specification

1. The abstract of the disclosure is objected to because a copy of the WIPO publication has been submitted. An abstract on a separate sheet of paper is required. Correction is required. See MPEP § 608.01(b).

Drawings

2. The drawings are objected to because they are not properly labeled as FIG. 1, FIG. 2, FIG. 3. Also, Figure 1 does not have the appropriate "Prior Art" label. Corrected drawing sheets in compliance with 37 CFR 1.121(d) are required in reply to the Office action to avoid abandonment of the application. Any amended replacement drawing sheet should include all of the figures appearing on the immediate prior version of the sheet, even if only one figure is being amended. The figure or figure number of an amended drawing should not be labeled as “amended.” If a drawing figure is to be canceled, the appropriate figure must be removed from the replacement sheet, and where necessary, the remaining figures must be renumbered and appropriate changes made to the brief description of the several views of the drawings for consistency. Additional replacement sheets may be necessary to show the renumbering of the remaining figures. Each drawing sheet submitted after the filing date of an application must be labeled in the top margin as either “Replacement Sheet” or “New Sheet” pursuant to 37 CFR 1.121(d). If the changes are not accepted by the examiner, the applicant will be notified and informed of any required corrective action in the next Office action. The objection to the drawings will not be held in abeyance.

Claim Rejections - 35 USC § 112

3. The following is a quotation of the first paragraph of 35 U.S.C. 112:

> The specification shall contain a written description of the invention, and of the manner and process of making and using it, in such full, clear, concise, and exact terms as to enable any person skilled in the art to which it pertains, or with which it is most nearly connected, to make and use the same and shall set forth the best mode contemplated by the inventor of carrying out his invention.

4. Claims 9-18 are rejected under 35 U.S.C. 112, first paragraph, as failing to comply with the enablement requirement. The claim(s) contains subject matter which was not described in the specification in such a way as to enable one skilled in the art to which it pertains, or with which it is most nearly connected, to make and/or use the invention. The term "computer readable medium" is not defined in the specification.

5. The following is a quotation of the second paragraph of 35 U.S.C. 112:

> The specification shall conclude with one or more claims particularly pointing out and distinctly claiming the subject matter which the applicant regards as his invention.

6. Claims 9-18 are rejected under 35 U.S.C. 112, second paragraph, as being indefinite for failing to particularly point out and distinctly claim the subject matter which applicant regards as the invention. The term computer-readable medium is not defined in the specification and therefore the claim is indefinite.

7. Claims 10-13 are further rejected under 35 U.S.C. 112, second paragraph, as being indefinite for failing to particularly point out and distinctly claim the subject matter which applicant regards as the invention. These Claims are further rejected because these claims represent improper dependence as claims 10-13, which are method claims, depend upon Claim 9, which is a product claim.

Claim Rejections - 35 USC § 101

8. 35 U.S.C. 101 reads as follows:

> Whoever invents or discovers any new and useful process, machine, manufacture, or composition of matter, or any new and useful improvement thereof, may obtain a patent therefor, subject to the conditions and requirements of this title.

9. Claims 9-18 are rejected under 35 U.S.C. 101 because the claimed invention is directed to non-statutory subject matter. The computer-readable medium is not defined in the specification as described above. Therefore, the computer program product of these claims is interpreted to be directed to Compute Software *per se*. Computer Software *per se* is considered functional descriptive material and therefore non-statutory when not claimed in combination with sufficient structure to render the claim statutory. Please see MPEP §2106.

Claim Rejections - 35 USC § 102

> (b) the invention was patented or described in a printed publication in this or a foreign country or in public use or on sale in this country, more than one year prior to the date of application for patent in the United States.

1. Claims 1-3, 6-15 and 18-20 are rejected under 35 U.S.C. 102(b) as being anticipated by US Patent 5,205,289 Palmer hereinafter Palmer.

Regarding These Claims Palmer teaches:

1. A method for processing a queue of messages, each message representing at least one request for an update to a database, the method comprising: browsing a message **(Palmer Column 4, Lines 47-51, “The server read message supplies a list of UIDs**

to the server, which gets the identified objects from disk or a server-managed buffer and returns them to the requester. A server may facilitate prefetching through extensions of the basic read function.); extracting from a browsed message an update request **(Column 4, Lines 52-53 "A requester may mark reads for either demand or prefetch processing.")**; and sending a pretend update request to a database management system (DBMS) responsible for the database which is to be updated, the pretend update request comprising an indication that directs the DBMS to not execute the update, but instead to prefetch data that will be required when a corresponding real update is requested **(e.g. Column 5, Lines 40-47, "The predictive cache subsystem can be configured as a separate service or as part of the client image. As illustrated in FIG. 2, a portion 25 of the client cache is allocated to the predictive cache. The cache manager 30 interfaces to the prophet 27 to decide what to prefetch and the order in which to replace cached objects. The prophet has two primary modes of operation: prediction an training.") and further Column 2, Lines 29-46, "In accordance with the present invention, a data processing system includes a predictive cache memory subsystem which allows objects accessed from secondary memory to be stored in cache memory. The subsystem includes a pattern memory for storing patterns of access to data objects. The stored patterns are generated by analysis of prior object accesses when the data processing system is processing in a like context. A prefetcher includes a predictor which responds to a current access pattern during operation of the data processing system to predict, from like object access patterns stored in pattern**

memory, accesses to objects. The predicted objects are prefetched and stored in cache memory. Though objects may take any form, such as in virtual memory management in an operating system, the invention has particular application to object oriented databases (OODBs)." It is inherent her that the "operating of a data processing system would include processing updating ("writing" in the disclosure) requests).

2. The method of claim 1, wherein the method comprises translating the pretend update request into a prefetch request, and prefetching required data **(Palmer Column 4, Lines 47-51, "The server read message supplies a list of UIDs to the server, which gets the identified objects from disk or a server-managed buffer and returns them to the requester. A server may facilitate prefetching through extensions of the basic read function.)**.

3. The method of claim 1, further comprising initiating a real update request by destructively getting a message from a queue comprising the update request, the real update request using prefetched data **(e.g. Column 6, Lines 46-50, "The state field indicates one of three states: promised (a read is in progress), prefetch (a prefetched object is resident in cache but as yet is unreferenced) and referenced (an object in cache has been referenced by the application program)." and further Column 2, Lines 29-46, "In accordance with the present invention, a data processing system includes a predictive cache memory subsystem which allows objects accessed from secondary memory to be stored in cache memory. The subsystem includes a pattern memory for storing patterns of access to data**

objects. The stored patterns are generated by analysis of prior object accesses when the data processing system is processing in a like context. A prefetcher includes a predictor which responds to a current access pattern during operation of the data processing system to predict, from like object access patterns stored in pattern memory, accesses to objects. The predicted objects are prefetched and stored in cache memory. Though objects may take any form, such as in virtual memory management in an operating system, the invention has particular application to object oriented databases (OODBs)." It is inherent her that the "operating of a data processing system would include processing updating ("writing" in the disclosure) requests).

6. The method of claim 2 wherein the prefetch request has a predetermined form and the method further comprises: retaining the predetermined form of the prefetch request **(e.g. Palmer Column 6, Lines 57-66, "FIG. 4 illustrates a pattern memory and a prediction formed from several unit patterns from memory. To implement a high speed yet reliable prediction strategy, patterns are stored in memory as units with a maximum pattern length k of, for example, fifty identifiers, though only nine are shown in the figure. In one implementation two thousand entries are allowed in the pattern memory for a context. In locating matching patterns, a sample window of current access objects is compared to the first objects of each pattern of pattern memory.")**; associating an identifier with the retained predetermined form in order that the predetermined form can be identified and used in subsequent performance of the real update request **(e.g. Palmer Column 6, Lines 57-66, "FIG. 4**

illustrates a pattern memory and a prediction formed from several unit patterns from memory. To implement a high speed yet reliable prediction strategy, patterns are stored in memory as units with a maximum pattern length k of, for example, fifty identifiers, though only nine are shown in the figure. In one implementation two thousand entries are allowed in the pattern memory for a context. In locating matching patterns, a sample window of current access objects is compared to the first objects of each pattern of pattern memory."); and returning the identifier in response to the pretend update request **(e.g. Column 7, Lines 11-14, "In the present embodiment, to search for the matching patterns, the predictor performs a binary search on the first column and locates any patterns having the object a in that column.")**.

7. The method of claim 1 further comprising: translating the pretend update request into a prefetch request in a predetermined form **(e.g. Palmer Column 6, Lines 57-66, "FIG. 4 illustrates a pattern memory and a prediction formed from several unit patterns from memory. To implement a high speed yet reliable prediction strategy, patterns are stored in memory as units with a maximum pattern length k of, for example, fifty identifiers, though only nine are shown in the figure. In one implementation two thousand entries are allowed in the pattern memory for a context. In locating matching patterns, a sample window of current access objects is compared to the first objects of each pattern of pattern memory.")**; associating the pretend update request with an identifier by the DBMS **(e.g. Palmer Column 6, Lines 57-66, "FIG. 4 illustrates a pattern memory and a prediction**

formed from several unit patterns from memory. To implement a high speed yet reliable prediction strategy, patterns are stored in memory as units with a maximum pattern length k of, for example, fifty identifiers, though only nine are shown in the figure. In one implementation two thousand entries are allowed in the pattern memory for a context. In locating matching patterns, a sample window of current access objects is compared to the first objects of each pattern of pattern memory."); receiving the identifier from the DBMS **(e.g. Palmer Column 6, Lines 57-66, "FIG. 4 illustrates a pattern memory and a prediction formed from several unit patterns from memory. To implement a high speed yet reliable prediction strategy, patterns are stored in memory as units with a maximum pattern length k of, for example, fifty identifiers, though only nine are shown in the figure. In one implementation two thousand entries are allowed in the pattern memory for a context. In locating matching patterns, a sample window of current access objects is compared to the first objects of each pattern of pattern memory.")**; and issuing the real update request by sending the identifier with the update request **(e.g. Column 2, Lines 29-46, "In accordance with the present invention, a data processing system includes a predictive cache memory subsystem which allows objects accessed from secondary memory to be stored in cache memory. The subsystem includes a pattern memory for storing patterns of access to data objects. The stored patterns are generated by analysis of prior object accesses when the data processing system is processing in a like context. A prefetcher includes a predictor which responds to a current access pattern**

during operation of the data processing system to predict, from like object access patterns stored in pattern memory, accesses to objects. The predicted objects are prefetched and stored in cache memory. Though objects may take any form, such as in virtual memory management in an operating system, the invention has particular application to object oriented databases (OODBs)." It is inherent her that the "operating of a data processing system would include processing updating ("writing" in the disclosure) requests).

8. The method of claim 1 further comprising informing a memory manager that the prefetched data used may be discarded from memory subsequent to the use of the prefetched data in the processing of a real update request **(Palmer e.g. Column 9, Lines 47-50, "If the object was not previously referenced, the system checks at 118 whether the entry state is that of a promise. If it is a promise, the system must wait for the returned A at 114, the object having been requested in a prior prefetch. The wait at 114 for the object A is ended with an I/O completion illustrated from 124 in FIG. 7C. K is the number of prefetched objects being returned with the I/O completion. K is checked at the beginning of the loop at 126. The object at the tail of the replacement list is evicted at 128. The prefetched object of lowest priority, that is at the tail of the prediction, is entered at the head of the ROT replacement list. Its state is set to prefetch and its pointer is set to an address in cache memory at 130. K is then decremented at 132 and again checked at 126.")**.

9. A computer program product comprising a computer readable medium having computer usable program code for pre-processing at a database management system (DBMS) of update requests to a database controlled by the DBMS, the computer program product comprising: computer usable program code for receiving an update request at the DBMS **(Palmer Column 4, Lines 47-51, "The server read message supplies a list of UIDs to the server, which gets the identified objects from disk or a server-managed buffer and returns them to the requester. A server may facilitate prefetching through extensions of the basic read function.)**; computer usable program code for receiving an indication at the DBMS indicating that the update request is a pretend update request that directs the DBMS to not execute an update request but instead to prefetch data for the update request **(Column 4, Lines 52-53 "A requester may mark reads for either demand or prefetch processing.")**; computer usable program code for translating the pretend update request into a prefetch request **(e.g. Column 5, Lines 40-47, "The predictive cache subsystem can be configured as a separate service or as part of the client image. As illustrated in FIG. 2, a portion 25 of the client cache is allocated to the predictive cache. The cache manager 30 interfaces to the prophet 27 to decide what to prefetch and the order in which to replace cached objects. The prophet has two primary modes of operation: prediction an training.")**; and computer usable program code for prefetching required data based on the prefetch request **(e.g. Column 5, Lines 40-47, "The predictive cache subsystem can be configured as a separate service or as part of the client image. As illustrated in FIG. 2, a portion 25 of the client cache is**

allocated to the predictive cache. The cache manager 30 interfaces to the prophet 27 to decide what to prefetch and the order in which to replace cached objects. The prophet has two primary modes of operation: prediction an training.").

10. The method of claim 9 further comprising receiving a real update request at the DBMS and executing the real update request using previously prefetched data **(Palmer e.g. Column 9, Lines 47-50, "If the object was not previously referenced, the system checks at 118 whether the entry state is that of a promise. If it is a promise, the system must wait for the returned A at 114, the object having been requested in a prior prefetch. The wait at 114 for the object A is ended with an I/O completion illustrated from 124 in FIG. 7C. K is the number of prefetched objects being returned with the I/O completion. K is checked at the beginning of the loop at 126. The object at the tail of the replacement list is evicted at 128. The prefetched object of lowest priority, that is at the tail of the prediction, is entered at the head of the ROT replacement list. Its state is set to prefetch and its pointer is set to an address in cache memory at 130. K is then decremented at 132 and again checked at 126.")**.

11. The method of claim 9 wherein the prefetch request has a predetermined form and the method further comprises: retaining the predetermined form of the prefetch request**(e.g. Palmer Column 6, Lines 57-66, "FIG. 4 illustrates a pattern memory and a prediction formed from several unit patterns from memory. To implement a high speed yet reliable prediction strategy, patterns are stored in memory as units with a maximum pattern length k of, for example, fifty identifiers, though**

only nine are shown in the figure. In one implementation two thousand entries are allowed in the pattern memory for a context. In locating matching patterns, a sample window of current access objects is compared to the first objects of each pattern of pattern memory."); associating an identifier with the retained predetermined form in order that the predetermined form can be identified and used in subsequent performance of the real update request **(e.g. Palmer Column 6, Lines 57-66, "FIG. 4 illustrates a pattern memory and a prediction formed from several unit patterns from memory. To implement a high speed yet reliable prediction strategy, patterns are stored in memory as units with a maximum pattern length k of, for example, fifty identifiers, though only nine are shown in the figure. In one implementation two thousand entries are allowed in the pattern memory for a context. In locating matching patterns, a sample window of current access objects is compared to the first objects of each pattern of pattern memory.")**; and returning the identifier in response to the pretend update request **(e.g. Palmer Column 6, Lines 57-66, "FIG. 4 illustrates a pattern memory and a prediction formed from several unit patterns from memory. To implement a high speed yet reliable prediction strategy, patterns are stored in memory as units with a maximum pattern length k of, for example, fifty identifiers, though only nine are shown in the figure. In one implementation two thousand entries are allowed in the pattern memory for a context. In locating matching patterns, a sample window of current access objects is compared to the first objects of each pattern of pattern memory.")**.

12. The method of claim 11 further comprising receiving the identifier with a real update request, and using the predetermined form associated with the identifier in performance of the real update request **(Palmer e.g. Column 9, Lines 47-50, "If the object was not previously referenced, the system checks at 118 whether the entry state is that of a promise. If it is a promise, the system must wait for the returned A at 114, the object having been requested in a prior prefetch. The wait at 114 for the object A is ended with an I/O completion illustrated from 124 in FIG. 7C. K is the number of prefetched objects being returned with the I/O completion. K is checked at the beginning of the loop at 126. The object at the tail of the replacement list is evicted at 128. The prefetched object of lowest priority, that is at the tail of the prediction, is entered at the head of the ROT replacement list. Its state is set to prefetch and its pointer is set to an address in cache memory at 130. K is then decremented at 132 and again checked at 126.")**.

13. The method of claim 9 further comprising informing a memory manager that the prefetched data may be discarded from memory subsequent to the use of the prefetched data in the processing of a real update request **(Palmer e.g. Column 9, Lines 47-50, "If the object was not previously referenced, the system checks at 118 whether the entry state is that of a promise. If it is a promise, the system must wait for the returned A at 114, the object having been requested in a prior prefetch. The wait at 114 for the object A is ended with an I/O completion illustrated from 124 in FIG. 7C. K is the number of prefetched objects being returned with the I/O completion. K is checked at the beginning of the loop at 126.**

The object at the tail of the replacement list is evicted at 128. The prefetched object of lowest priority, that is at the tail of the prediction, is entered at the head of the ROT replacement list. Its state is set to prefetch and its pointer is set to an address in cache memory at 130. K is then decremented at 132 and again checked at 126.").

14. A computer program product comprising a computer readable medium having computer usable program code for processing a queue of messages, each message representing at least one request for an update to a database, the computer program product comprising: computer usable program code for browsing an unexecuted message **(Palmer Column 4, Lines 47-51, "The server read message supplies a list of UIDs to the server, which gets the identified objects from disk or a server-managed buffer and returns them to the requester. A server may facilitate prefetching through extensions of the basic read function.)**; computer usable program code for extracting an update request from an unexecuted message **(Column 4, Lines 52-53 "A requester may mark reads for either demand or prefetch processing.")**; and computer usable program code for translating the update request into a query request to prefetch data for the unexecuted update request**(e.g. Column 5, Lines 40-47, "The predictive cache subsystem can be configured as a separate service or as part of the client image. As illustrated in FIG. 2, a portion 25 of the client cache is allocated to the predictive cache. The cache manager 30 interfaces to the prophet 27 to decide what to prefetch and the order in which to replace**

cached objects. The prophet has two primary modes of operation: prediction an training.").

15. The computer program product of claim 14 further comprising computer usable program code for initiating a real update request by destructively getting a message from a queue comprising the update request, the real update request using prefetched data **(e.g. Column 6, Lines 46-50, "The state field indicates one of three states: promised (a read is in progress), prefetch (a prefetched object is resident in cache but as yet is unreferenced) and referenced (an object in cache has been referenced by the application program).")**.

18. The computer program product of claim 14 further comprising computer usable program code for informing a memory manager that the prefetched data used may be discarded from memory subsequent to the use of the prefetched data in the processing of a real update request **(Palmer e.g. Column 9, Lines 47-50, "If the object was not previously referenced, the system checks at 118 whether the entry state is that of a promise. If it is a promise, the system must wait for the returned A at 114, the object having been requested in a prior prefetch. The wait at 114 for the object A is ended with an I/O completion illustrated from 124 in FIG. 7C. K is the number of prefetched objects being returned with the I/O completion. K is checked at the beginning of the loop at 126. The object at the tail of the replacement list is evicted at 128. The prefetched object of lowest priority, that is at the tail of the prediction, is entered at the head of the ROT replacement list. Its state is set to**

prefetch and its pointer is set to an address in cache memory at 130. K is then decremented at 132 and again checked at 126.").

19. A computer implemented method for facilitating database performance by pre-processing update requests to a database management system (DBMS) for a queue of messages, comprising: executing a computer program product configured to: receive an update request at the DBMS **(Palmer Column 4, Lines 47-51, "The server read message supplies a list of UIDs to the server, which gets the identified objects from disk or a server-managed buffer and returns them to the requester. A server may facilitate prefetching through extensions of the basic read function.)**; receive an indication at the DBMS indicating that the update request is a pretend update request that directs the DBMS to not execute the update but instead to prefetch data for the update request **(Column 4, Lines 52-53 "A requester may mark reads for either demand or prefetch processing.")**; translate the pretend update request into a prefetch request; prefetch required data based on the prefetch request **(e.g. Column 5, Lines 40-47, "The predictive cache subsystem can be configured as a separate service or as part of the client image. As illustrated in FIG. 2, a portion 25 of the client cache is allocated to the predictive cache. The cache manager 30 interfaces to the prophet 27 to decide what to prefetch and the order in which to replace cached objects. The prophet has two primary modes of operation: prediction an training.")**; and receiving a real update request at the DBMS **(Column 2, Lines 29-46, "In accordance with the present invention, a data processing system includes a predictive cache memory subsystem which allows objects accessed from**

secondary memory to be stored in cache memory. The subsystem includes a pattern memory for storing patterns of access to data objects. The stored patterns are generated by analysis of prior object accesses when the data processing system is processing in a like context. A prefetcher includes a predictor which responds to a current access pattern during operation of the data processing system to predict, from like object access patterns stored in pattern memory, accesses to objects. The predicted objects are prefetched and stored in cache memory. Though objects may take any form, such as in virtual memory management in an operating system, the invention has particular application to object oriented databases (OODBs)." It is inherent her that the "operating of a data processing system would include processing updating ("writing" in the disclosure) requests); and executing the real update request using the prefetched data **(Column 2, Lines 29-46, "In accordance with the present invention, a data processing system includes a predictive cache memory subsystem which allows objects accessed from secondary memory to be stored in cache memory. The subsystem includes a pattern memory for storing patterns of access to data objects. The stored patterns are generated by analysis of prior object accesses when the data processing system is processing in a like context. A prefetcher includes a predictor which responds to a current access pattern during operation of the data processing system to predict, from like object access patterns stored in pattern memory, accesses to objects. The predicted objects are prefetched and stored in cache memory. Though objects may take any form, such as in virtual**

memory management in an operating system, the invention has particular application to object oriented databases (OODBs)." It is inherent her that the "operating of a data processing system would include processing updating ("writing" in the disclosure) requests).

20. The computer implemented method of claim 19 further comprising informing a memory manager that the prefetched data may be discarded from memory subsequent to the use of the prefetched data in the processing of a real update request **(Palmer e.g. Column 9, Lines 47-50, "If the object was not previously referenced, the system checks at 118 whether the entry state is that of a promise. If it is a promise, the system must wait for the returned A at 114, the object having been requested in a prior prefetch. The wait at 114 for the object A is ended with an I/O completion illustrated from 124 in FIG. 7C. K is the number of prefetched objects being returned with the I/O completion. K is checked at the beginning of the loop at 126. The object at the tail of the replacement list is evicted at 128. The prefetched object of lowest priority, that is at the tail of the prediction, is entered at the head of the ROT replacement list. Its state is set to prefetch and its pointer is set to an address in cache memory at 130. K is then decremented at 132 and again checked at 126.")**.

Claim Rejections - 35 USC § 103

2. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

3. Claims 4,5, 16 and 17 are rejected under 35 U.S.C. 103(a) as being unpatentable over US Patent 5,205,289 Palmer hereinafter Palmer in view of US Patent 6,092,154 Curtis et al hereinafter Curtis

4. Regarding these Claims, Palmer and Curtis teach:

4. Palmer teaches: The method of claim 3. However, Palmer does not teach: wherein initiating a real update request is performed by a master thread and browsing a message is performed by one or more read ahead threads However, this limitation it taught by Curtis **(Curtis e.g. Column 6, Lines 42-53, "Generic threads may be either sequential or predictive. Sequential threads are useful in applications where a path through the data defined by the thread is set in advance. Predictive threads are used to specify a random accessed list of samples, sorted in the order of probability of access. Predictive threads allow access to any sample in a list where the data is defined in its native form (for example, a video frame or an audio chunk). Predictive threads also allow the cache size to be specified at creation time. Finally, predictive threads allow the drive to reorder the thread based on recent host data requests." Or Column 2, Lines 55-58 "In other words, the storage device continues to locate data on the list and store it in the buffer**

while the host sends commands to retrieve the data that it needs from the buffer.") In addition it would have been obvious at the time of the invention to combine the teachings of Palmer with the threading tools of Curtis as Both systems are directed to pre-fetching of data in a data processing system, and The threads of Curtis would allow the message pre-fetching system of Palmer to be multithreaded.

5. Curtis further teaches: The method of claim 4, wherein processing of the master thread is maintained behind the read ahead thread by a predetermined amount **(Curtis e.g. Column 6, Lines 54-59, "In general, threads may be either dynamic or persistent. Dynamic threads must be created by the host application. Persistent threads, on the other hand, are saved on the storage media and restored in cache after each power cycle or reset. Persistent threads define system or application specific data parameters that are constantly accessed, such as file allocation tables.")**.

16. Palmer teaches: The computer program product of claim 15. Palmer does not teach: further comprising computer usable program code wherein initiating a real update request is performed by a master thread and browsing a message is performed by one or more read ahead threads. However, this limitation is taught by Curtis. **(Curtis e.g. Column 6, Lines 42-53, "Generic threads may be either sequential or predictive. Sequential threads are useful in applications where a path through the data defined by the thread is set in advance. Predictive threads are used to specify a random accessed list of samples, sorted in the order of probability of access. Predictive threads allow access to any sample in a list where the data is defined**

in its native form (for example, a video frame or an audio chunk). Predictive threads also allow the cache size to be specified at creation time. Finally, predictive threads allow the drive to reorder the thread based on recent host data requests." Or Column 2, Lines 55-58 "In other words, the storage device continues to locate data on the list and store it in the buffer while the host sends commands to retrieve the data that it needs from the buffer.") In addition it would have been obvious at the time of the invention to combine the teachings of Palmer with the threading tools of Curtis as Both systems are directed to pre-fetching of data in a data processing system, and The threads of Curtis would allow the the message pre-fetching system of Palmer to be multithreaded.

17. Curtis further teaches: The computer program product of claim 16 further comprising computer usable program code wherein processing of the master thread is maintained behind the read ahead thread by a predetermined amount **(Curtis e.g. Column 6, Lines 54-59, "In general, threads may be either dynamic or persistent. Dynamic threads must be created by the host application. Persistent threads, on the other hand, are saved on the storage media and restored in cache after each power cycle or reset. Persistent threads define system or application specific data parameters that are constantly accessed, such as file allocation tables.")**.

Conclusion

Any inquiry concerning this communication or earlier communications from the examiner should be directed to MATTHEW J. BROPHY whose telephone number is . The examiner can normally be reached on Monday-Thursday 8:00AM-5:00 PM EST.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Wei Zhen can be reached on (571) 272-3708. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free). If you would like assistance from a USPTO Customer Service Representative or access to the automated information system, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

MJB

2/28/2008

/Ted T. Vo/
Primary Examiner, Art Unit 2191